

Serial No: 09/683,383

IN THE CLAIMS:

Please amend claims 1, 4-6 and 10, and add new claims 11-16 as follows:

1 Claim 1. (currently amended) A method for operating an out-of-order
2 processor comprised of an instruction pipeline, the method comprising the
3 steps of:
4 for detection of a dependency, determining for each current instruction
5 involved in a renaming process that a logic target address of one or more
6 instructions is not the same as a logic source address of said current
7 instruction, said one or more instructions being stored in a temporary buffer
8 associated with a pipeline process downstream of the current instruction;
9 generating a no-dependency signal associated with said current
10 instruction; and
11 bypassing a portion of the instruction pipeline for the current
12 instruction if the no-dependency signal is active; and
13 addressing a mapping-table-entry of a mapping table with a logical
14 source register address of said current instruction thus determining the
15 mapped physical target register address.

1 Claim 2. (previously presented) The method according to claim 1 in
2 which the step of generating a no-dependency signal comprises the steps of:
3 comparing a plurality of logic target register addresses and the logic
4 source register address of the current instruction;
5 in case the logic target register addresses and the logic source
6 register address match, setting the no-dependency signal to not active; and
7 generating a dependency signal for the respective source register.

1 Claim 3. (previously presented) The method according to claim 1
2 further comprising the step of evaluating 'valid'-bits of speculative target
3 registers stored in a storage associated with speculatively calculated
4 instruction result data to generate the no-dependency signal.

1 Claim 4. (currently amended) The method according to claim 1 further
2 comprising ~~the step of applying the method to a mapping table based renaming~~
3 ~~scheme comprising the steps of:~~
4 ~~addressing a mapping-table-entry of a mapping table with a logical~~
5 ~~source register address of said current instruction thus determining the~~
6 ~~mapped physical target register address;~~
7 reading a committed-status flag in said entry;
8 comparing the logic target register address and the logic source

Serial No: 09/683,383

9 register address of the current instruction in case the no-dependency signal
10 is not active; and
11 generating a dependency signal for the respective source register.

1 Claim 5. (currently amended) The method according to claim 1 further
2 ~~comprising the step of applying the method to a mapping table based renaming~~
3 ~~scheme comprising the steps of:~~
4 ~~addressing a mapping table entry with a logical source register address~~
5 ~~of said current instruction thus determining the mapped physical target~~
6 ~~register address;~~
7 reading a committed-status flag in said entry;
8 comparing the logic target register address and the logic source
9 register address of the current instruction;
10 in case the logic target register address and the logic source register
11 address match, setting the no-dependency signal to not active; and
12 generating a dependency-signal for the respective source register.

1 Claim 6. (currently amended) A processing system having means for
2 executing a readable machine language, said readable machine language
3 comprises:
4 a first computer readable code embodied in tangible media for, the
5 detection of a dependency, determining for each current instruction involved
6 in a renaming process that a logic target address of one or more instructions
7 stored in a temporary buffer associated with a pipeline process downstream of
8 the current instruction is not the same as a logic source address of said
9 current instruction,
10 a second computer readable code embodied in tangible media for
11 generating a no-dependency signal associated with said current instruction,
12 and
13 a third computer readable code embodied in tangible media for bypassing
14 a portion of the instruction pipeline for the current instruction if the no-
15 dependency signal is active, and
16 a fourth computer readable code embodied in tangible media for
17 addressing a mapping-table-entry of a mapping table with a logical source
18 register address of said current instruction thus determining the mapped
19 physical target register address.

1 Claim 7. (previously presented) The processing system according to
2 claim 6 in which in case of a content-addressable memory (CAM)-based renaming
3 scheme the first computer readable code for determining the dependency of a

Serial No: 09/683,383

4 current instruction comprises a compare logic in which all instructions to be
5 checked for dependency are involved and an OR gate coupled with the compare
6 logic.

1 Claim 8. (previously presented) The processing system according to
2 claim 7 further comprising a plurality of AND gates the input of which
3 comprises a target register 'valid bits' signal and a respective compare
4 logic output signal.

1 Claim 9. (previously presented) The processing system according to
2 claim 6 in which the case of a mapping-table-based renaming scheme each
3 mapping table entry comprises an additional instruction-committed flag, and
4 the first computer readable code for determining the dependency of a current
5 instruction comprises a logic for ANDing a target register 'valid bits'
6 signal in which all instructions to be checked for dependency are involved
7 and an OR gate coupled with the logic.

1 Claim 10. (currently amended) A computer system having an out-of-order
2 processing system, said computer system executes a readable machine language,
3 said readable machine language comprises:

4 a first computer readable code embodied in tangible media for, the
5 detection of a dependency, determining for each current instruction involved
6 in a renaming process that a logic target address of one or more instructions
7 stored in a temporary buffer associated with a pipeline process downstream of
8 the current instruction is not the same as a logic source address of said
9 current instruction,

10 a second computer readable code embodied in tangible media for
11 generating a no-dependency signal associated with said current instruction,

12 a third computer readable code embodied in tangible media for assigning
13 an entry in the temporary buffer to the logic source address of said current
14 instruction if the no-dependency signal is not active; and

15 a fourth computer readable code embodied in tangible media for issuing
16 the instruction operand data to an instruction execution unit without
17 assigning the entry in the temporary buffer to the logic source address of
18 said current instruction if the no-dependency signal is active; and

19 a fifth computer readable code embodied in tangible media for
20 addressing a mapping-table-entry of a mapping table with a logical source
21 register address of said current instruction thus determining the mapped
22 physical target register address.

Serial No: 09/683,383

1 Claim 11. (new) The method according to claim 1, further comprising
2 partitioning the current instruction into a reorder buffer and an
3 architectural register array, the reorder buffer configured to store
4 speculative results of the current instruction and the architectural register
5 array configured to store an architectural state of the processor.

1 Claim 12. (new) The method according to claim 11, wherein the no-
2 dependency signal is logically ANDed with a plurality of validity bits
3 indicating data is available at the reorder buffer and the architectural
4 register array.

1 Claim 13. (new) The processing system according to claim 6, further
2 comprising a fifth computer readable code embodied in tangible media for
3 partitioning the current instruction into a reorder buffer and an
4 architectural register array, the reorder buffer configured to store
5 speculative results of the current instruction and the architectural register
6 array configured to store an architectural state of the processor.

1 Claim 14. (new) The processing system according to claim 13, wherein
2 the no-dependency signal is logically ANDed with a plurality of validity bits
3 indicating data is available at the reorder buffer and the architectural
4 register array.

1 Claim 15. (new) The computer system according to claim 10, further
2 comprising a sixth computer readable code embodied in tangible media for
3 partitioning the current instruction into a reorder buffer and an
4 architectural register array, the reorder buffer configured to store
5 speculative results of the current instruction and the architectural register
6 array configured to store an architectural state of the processor.

1 Claim 16. (new) The computer system according to claim 15, wherein the
2 no-dependency signal is logically ANDed with a plurality of validity bits
3 indicating data is available at the reorder buffer and the architectural
4 register array.